Department of
Information Engineering
and Computer Science
DISI

UNIVERSITY
OF TRENTO - Italy

DISI - Via Sommarive, 5 - 38123 POVO, Trento - Italy
http://disi.unitn.it

PRACTICAL DATA PREDICTION FOR REAL-WORLD WIRELESS SENSOR NETWORKS

Usman Raza, Alessandro Camerra, Amy L. Murphy, Themis Palpanas, and Gian Pietro Picco

April 2014

Technical Report # DISI-14-004

Practical Data Prediction for Real-World Wireless Sensor Networks

Usman Raza, Alessandro Camerra, Amy L. Murphy, Themis Palpanas, and Gian Pietro Picco

Abstract—Data prediction is proposed in wireless sensor networks (WSNs) to extend the system lifetime by enabling the sink to determine the data sampled, within some accuracy bounds, with only minimal communication from source nodes. Several theoretical studies clearly demonstrate the tremendous potential of this approach, able to suppress the vast majority of data reports at the source nodes. Nevertheless, the techniques employed are relatively complex, and their feasibility on resource-scarce WSN devices often not ascertained. More generally, the literature lacks reports from real-world deployments, quantifying the overall system-wide lifetime improvements determined by the interplay of data prediction with the underlying network. These two aspects, feasibility and system-wide gains, are key in determining the *practical* usefulness of data prediction in real-world WSN applications.

In this paper, we describe Derivative-Based Prediction (DBP), a novel data prediction technique much simpler than state-of-the-art ones. Evaluation with real data sets from diverse WSN deployments shows that DBP often performs better than the competition, with data suppression rates up to 99% and good prediction accuracy. However, experiments with real WSNs show that, when the network stack is taken into consideration, DBP only *triples* lifetime—a remarkable result per se, but a far cry from the data suppression rates above. To fully achieve the energy savings enabled by data prediction, the data and network layers must be jointly optimized. In our testbed, a simple tuning of the MAC and routing stack, taking into account the operation of DBP, yields a remarkable seven-fold lifetime improvement w.r.t. the mainstream periodic reporting.

Index Terms—Wireless sensor networks, data prediction, time series forecasting, energy efficiency, network protocols

1 Introduction

Wireless sensor networks (WSNs) provide the flexibility of untethered sensing, but pose the challenge of achieving extended lifetime with a limited energy budget, often provided by batteries. In this respect, it is well-known that communication causes the biggest energy drain. This is unfortunate, given that the ability to report sensed data motivates the use of WSNs in several pervasive computing applications.

- *U. Raza is with the Bruno Kessler Foundation, Italy and the University of Trento, Italy. E-mail: raza@fbk.eu*
- *A. Camerra is with IBM, Italy. E-mail: a.camerra@studenti.unitn.it*
- *A.L. Murphy is with the Bruno Kessler Foundation, Italy. E-mail: murphy@fbk.eu*
- *T. Palpanas is with the Paris Descartes University, France and the University of Trento, Italy. E-mail: themis@mi.parisdescartes.fr*
- *G.P. Picco is with the University of Trento, Italy. E-mail: gianpietro.picco@unitn.it.*

An approach to reduce communication without compromising data quality is to *predict* the trend followed by the data being sensed, an idea at the core of many techniques [1]. This data prediction approach[1] is applicable when data is reported periodically—the common case in many pervasive computing applications. In these cases, a model of the data trend can be computed locally to a node. This model constitutes the information being reported to the data collection sink, replacing several raw samples. As long as the locally-sensed data are compatible with the model prediction, no further communication is needed: only when the sensed data deviates from the model, must the latter be updated and sent to the sink. Section 2 formulates the data prediction problem in more detail.

The aforementioned approach is well-known, and has been proposed by several works we concisely survey in Section 6. Nevertheless, to the best of our knowledge none of these works has been verified in practice, in a real-world WSN deployment. On one hand, the techniques employed are relatively complex, and their effectiveness typically evaluated based on implementations in high-level languages (e.g., Java) on mainstream hardware platforms. Therefore, their feasibility on resource-scarce WSN devices remains not ascertained. Moreover, the works in the literature typically evaluate the gains only in terms of messages suppressed w.r.t. a standard approach sending all samples. This data-centric view, however, is quite optimistic. WSNs consume energy not only when transmitting and receiving data, but also in several *continuous* control operations driven by the network layer protocols, e.g., when maintaining a routing tree for data collection, or probing for ongoing communication at the MAC layer.

Therefore, the true question, currently unanswered by the literature, is to what extent the theoretical savings enabled by data prediction are actually observable *in practice*, i.e., *i)* on the resource-scarce devices typical of WSNs, and *ii)* when the application and

1. The techniques discussed here are known under various names, including *time-series forecasting*, *data modeling*, *prediction-based data collection*, and *model-driven data acquisition*. Although in a preliminary version of this paper [2] we used the last term, in this paper we resort to the more intuitive *data prediction*.

network stacks are combined in a single, deployed system. The goal of this paper is to provide an answer to this question, through the following contributions:

- We propose *Derivative-Based Prediction* (DBP), a novel data prediction technique compatible with applications requiring hard guarantees on data quality. DBP, described in Section 3, predicts the trend of data measured by a sensor node, and is considerably simpler than existing methods, making it amenable for resource-scarce WSNs, as witnessed by our TinyOS implementation for the popular TelosB motes.
- We perform an extensive experimental evaluation of DBP against state-of-the-art data prediction techniques, based on 7 diverse real-world data sets with more than 13 million data points in total. The results demonstrate the effectiveness of DBP, which often performs better than the competition by suppressing up to 99% of data transmissions while maintaining data quality within the required application tolerances.
- We describe the first[2] study of the interaction of data prediction with WSN network protocols, directly comparing the theoretical application-level gains against the practical, system-wide ones. We evaluate the performance of a staple network stack consisting of CTP [3] and Box-MAC [4], both in a 40-node indoor testbed and in a real application setting, a road tunnel [5]. Our results show that the gains attained in practice lead to a three-fold WSN lifetime improvement, which is a significant achievement in absolute terms, but dramatically lower than those derived in theory.
- We explore the potential of cross-layer network stack optimizations to further improve the lifetime of WSN nodes running DBP. In our tunnel application, we show how a careful, yet simple, joint parameter tuning of the MAC and routing layers reduces the network control overhead considerably, without affecting the DBP operation, and yields a remarkable seven-fold lifetime improvement w.r.t. the standard periodic reporting.

The paper ends with the concluding remarks of Section 7, underlining the further lifetime improvements and enhanced reliability that can be attained by a WSN network stack expressly designed to work in conjunction with data prediction techniques.

2 Problem Formulation

Data collection is a fundamental functionality of many WSN applications, and is commonly implemented by nodes periodically taking sensor measurements and reporting the corresponding samples to a data sink.

The premise of applying data prediction is that communication can be significantly reduced by avoiding transmission of each raw sample to the sink. This is achieved by using a model to estimate the sensed values, and by communicating with the sink only when changes in the sampled data render the model no longer able to accurately describe them.

In more detail, the data prediction strategy, applied on each node, involves the following general steps. The sensor builds a model of its data based on some initial, observed values, and transmits the model to the sink. From that point on, the sink operates on the assumption that the data observed by the sensor are within the value tolerance of the data predicted by the model. At the same time, the node is also using the model to predict its own sensor data, and compares the predicted values with those actually observed. If their difference is within the error tolerance, no further action is required. Otherwise, the sensor builds a new model and transmits it to the sink.

To enable this strategy, the application running at the sink must allow for a small tolerance in the accuracy of the reported data—an assumption that holds in the majority of WSN applications. In contrast with the ideal requirement of the sink obtaining *exact* values in *all* data reports, the correctness of these applications is unaffected as long as

1) the reported values match *closely* the exact ones;
2) inaccurate values occur only *occasionally*.

In other words, deviations from the exact reports are acceptable, as long as their extent in terms of difference in *value* and *time interval* during which the deviation occurs are small enough. In this paper we only consider non-probabilistic techniques that can provide *hard guarantees* on their predictions, a requirement for several real-world applications. We capture these assumptions with the following definitions:

- Let V_i be an exact measurement taken at time t_i. The *value tolerance* is defined by the maximum relative and absolute errors acceptable, $\varepsilon_V = (\epsilon^{rel}, \epsilon^{abs})$. From the application perspective, reading a value V_i becomes equivalent to reading any value $\hat{V}_i$ in the range R_V defined by the maximum error, $\hat{V}_i \in R_V = [V_i - \epsilon, V_i + \epsilon]$, where $\epsilon = \max\{\frac{V_i}{100}\epsilon^{rel}, \epsilon^{abs}\}$. In other words, the application considers a value $\hat{V}_i \in R_V$ as *correct*.
- Let $T = |t_j - t_k|$ be a time interval, and $\hat{V}_T = \{\hat{V}_j, \ldots, \hat{V}_k\}$ the set of values reported to the application during T. The *time tolerance* ε_T is the maximum acceptable value of T such that *all* the values reported in this interval are incorrect, i.e., $\hat{V}_i \notin R_V, \forall\, \hat{V}_i \in \hat{V}_T$.

The intuition behind these is shown in Fig. 1. Data prediction aims to suppress as many data reports from the WSN nodes as possible, while ensuring that the data used by the application at the sink is within the value and time tolerances ε_V and ε_T specified as part of the requirements. The use of both absolute and relative errors in the value tolerance is dictated by the requirements of several applications in which values

2. A preliminary version of this paper appeared in [2].

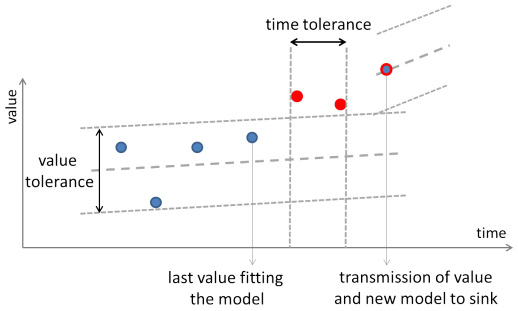


Fig. 1. Value and time tolerance.

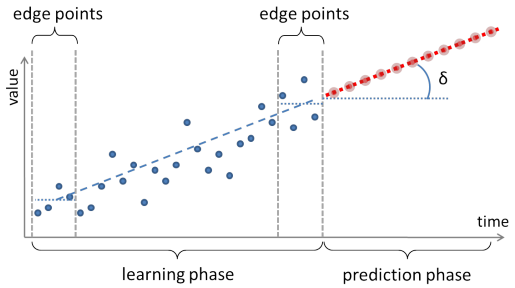


Fig. 2. Derivative-based Prediction.

can be very small but also very large. Our evaluation in Section 4 provides one concrete example, the TUNNEL application, where this problem is relevant. If only the absolute error ϵ^{abs} is used, it is difficult to set it in a way meaningful for both very small and very large values. On the other hand, a relative error ϵ^{rel} is often not very useful in the case of very small values, when the quantities at stakes are negligible. Using the maximum between relative and absolute error as value tolerance allows one to specify error in relative terms, and at the same time set an absolute threshold beyond which variations can be ignored.

3 Derivative-Based Prediction (DBP)

The idea behind the DBP technique is to use a simple model that can effectively capture the main data trends, and to compute this model in a way that is resilient to the noise inherent in the data. DBP is based on the observation that the trends of sensed values in short and medium time intervals can be accurately approximated using a linear model. Even though this idea has appeared in previous works, there is a key difference to our approach: while previous studies compute models that aim to reduce the approximation error to the *data points* in the recent past, DBP aims at producing models that are consistent with the *trends* in the recently-observed data.

Fig. 2 provides an illustration of DBP. Initialization consists of a *learning phase*, gathering enough data to produce the first model. The learning phase involves m data points; the first and the last l we call *edge points*. The model is linear, computed as the slope δ of the segment connecting the average values over the l edge points at the beginning and end of the learning phase. This computation resembles the calculation of the derivative, hence the name *Derivative-Based Prediction*. Interestingly, the computation of this prediction is not only very simple, and therefore appealing for implementation on resource-scarce nodes, it also mitigates the problem of noise and outliers.

The first DBP model generated is sent to the sink. From that time on, each node buffers a sliding window of the last m data points sampled from its sensor. Upon sampling a point, the "true" value sensed is compared to the one "predicted" by DBP according to the current model, i.e., following the slope δ. If the sensor reading is within the value tolerance ε_V w.r.t. the model, no action is required: the sink automatically generates a new value that is an acceptable approximation of the real one. Otherwise, if the readings continuously deviate from the model for more than ε_T time units, a new model must be recomputed, using the m buffered data points, and sent to the sink.

Implementation Considerations. As our final goal is to deploy DBP on real WSN nodes, the complexity and resource requirements (i.e., memory and CPU) of the implementation are very important, as these devices are typically not equipped with large memory or powerful CPUs. For instance, the popular TelosB motes used by the majority of WSN deployments reported in the literature, including the one about adaptive lighting in road tunnels [5] we illustrate in the next section, are equipped with only 48 KB of code memory, 10 KB of RAM, and an 8 MHz microcontroller suited for integer operations only.

In this respect, DBP is very efficient, involving only one subtraction, two summations, and two divisions to build the model, and a single summation for predicting the next value. Our DBP implementation in TinyOS requires only 50 lines of low-level[3] code, without the need to include any external libraries, or to use floating point arithmetic. As node memory is limited, eliminating the floating point arithmetic module is highly desirable. Further, our DBP implementation uses only 108 B of RAM, leaving almost all of the data memory to the application and the network stack.

In contrast, other state-of-the-art techniques (e.g., those considered in Section 4) employ mathematical libraries for solving linear equations with 2 and 3 unknowns to compute an autoregressive model (SAF), and a linear (PLA and SAF) and quadratic polynomial (POR) regression using least squares minimization. The above requirements render these approaches considerably more resource-intensive.

4 Application-Level Evaluation

This section analyzes the ability of our data prediction technique, DBP, to reduce the amount of data that

3. The equivalent Java routine consists of only 8 lines of code.

TABLE 1
Datasets characteristics and evaluation parameters

Application	Dataset	Sampling Period	Nodes	Samples	Error Tolerance (ϵ^{rel},ϵ^{abs})	ε_T	Learning Window (m)
TUNNEL	Light	30 s	40	5,414,400	(5%, 25 counts)	2	20
SOIL	Air Temperature	10 minutes	10	225,360	(5%, 0.5°C)	2	6
	Soil Temperature	10 minutes	4	77,904	(5%, 0.5°C)	2	6
INDOOR	Humidity	31 s	54	2,303,255	(5%, 1%)	2	20
	Light	31 s	54	2,303,255	(5%, 15 lx)	2	20
	Temperature	31 s	54	2,303,255	(5%, 0.5°C)	2	20
WATER	Chlorine	5 minutes	166	715,460	(5%, 0.0001)	2	6

must be transmitted to the sink. This is notably different from the system-wide energy savings enabled by such data suppression, which we analyze in Section 5.

We evaluate and compare data prediction techniques using the *suppression ratio*

$$SR = 1 - \frac{\text{\# messages generated with prediction}}{\text{\# messages generated without prediction}}$$

as our primary performance metric. SR directly measures the fraction of application-layer messages whose reporting can be avoided: the higher the value of SR, the more effectively a technique is performing.

4.1 Applications and Datasets

Our evaluation is based on 7 datasets from 4 different applications, described next, which cover a variety of data variation patterns, sampling periods, and number of nodes. Table 1 outlines the main characteristics of the datasets. Moreover, it reports the error tolerance we set as a requirement, based on the *real* tolerance used in the application as obtained by its designers or, in its absence, by considering the nature of the application. Finally, we report the learning window m, which is a characteristic not only of DBP but of all approaches, and is set at the same value for the sake of comparison.

These datasets contain *real* collected data, which was subject to losses on the wireless channel or to hardware failures of some nodes. This is different from an online application of data prediction, where each node has a perfect record of the sensed values, as they are being sampled on the node itself. Therefore, before running our evaluation, we reconstruct a perfect data series for each node by removing duplicates and interpolating for missing values, in the line of similar evaluations found in the literature [6].

Adaptive Lighting in Road Tunnels (TUNNEL). Our first case study involves a real-world WSN application, deployed in a road tunnel to acquire light readings [5]. The values are relayed in multi-hop to a gateway, and from there to a Programmable Logic Controller (PLC) that closes the control loop by setting the intensity of the lamps inside the tunnel. In contrast with the state of the art in tunnels, where light intensity is pre-set based on the current date and time, or at best determined by the external conditions, this closed-loop adaptive lighting system maintains optimal light levels by considering the *actual* conditions inside the tunnel. This increases safety, and enables considerable energy savings.

WSNs are an asset in this scenario, as the nodes can be placed at arbitrary points along the tunnel, not only where power and networking cables can reach. This drastically reduces installation and maintenance costs, and makes WSNs particularly appealing for existing tunnels, where changes to the infrastructure should be minimized. The downside to such flexibility is the reliance on an autonomous energy source. Nevertheless, battery costs are minimal and the replacement process can be easily combined with regularly-planned tunnel maintenance.

Fig. 3 shows the placement of WSN nodes inside our 260 m-long, two-way, two-lane tunnel. Overall, 40 nodes are split evenly between the tunnel walls and placed at a height of 1.70 m, compatible with legal regulations. Their data reports are collected by a gateway, installed 2 m from the entrance. Each node is functionally equivalent to a TelosB mote [7], augmented with a sensor board equipped with 4 ISL29004 digital light (illuminance) sensors. This setup is similar to the one reported in [5], where we detail and evaluate the operational WSN-based, closed-loop adaptive lighting system. In this paper we use a different application and network stack, and compare data prediction techniques against the baseline represented by the aforementioned periodic reporting of all samples.

The dataset we use contains the light readings reported every 30 s from each node for 47 days, for a total of $5,414,400$ measurements—the largest among the datasets we consider here. To establish the proper value and time tolerances, we consulted the lighting engineers who designed the control algorithm that establishes the lamp levels. By taking into consideration the inherent error of illuminance sensors, they determined a value tolerance $\varepsilon_V = (5, 25)$, i.e., values generated by the model can differ from the raw sensor reading by at most 5% or 25 counts, the latter corresponding approximately to 15 lx. Based on the application requirement that lamp levels must be adjusted slowly to minimize the effects of changes on

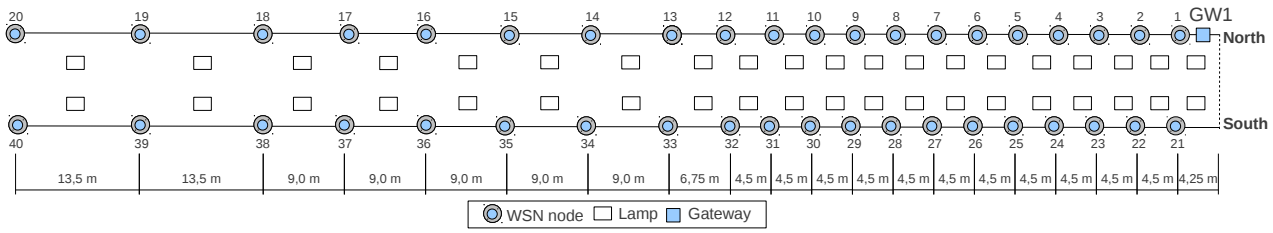


Fig. 3. Physical placement of WSN nodes in TUNNEL.

the drivers. They also identified a time tolerance of one minute. For convenience, we express ε_T in terms of the 30 s reporting intervals of the application; a one-minute time tolerance corresponds to $\varepsilon_T = 2$. We further establish the number of values in the learning phase of data prediction techniques to be $m = 20$, corresponding to a period of 10 minutes.

Soil Ecology (SOIL). Our second case study uses data originating in the Life Under Your Feet (LUYF) project [8]. LUYF brings biologists and WSN experts together to study the soil micro-climate in different forests of Maryland. As environmental conditions affect the activities and behavior of plants, micro-organisms and insects in the soil, a large WSN offers accurate, fine-grained spatial and temporal data, collected without being intrusive to the living creatures. Our study uses the soil and air temperature datasets collected in an urban forest in Baltimore, over a period of 225 days between September 2005 and July 2006. The soil and air temperatures are measured on the surface of earth and inside the box of the node, respectively. Despite their commonalities such as the presence of a diurnal cycle, the two temperature datasets exhibit distinctly different variation patterns. The soil temperature varies gradually over time with changes lagging those of air temperature by several hours due to a large inertia caused by the soil [9]. We determined the value tolerance for the temperature data sets in consultation with the soil scientists of LUYF project. Interestingly, the scientists are not interested in the temperature itself, rather in the production of CO_2 due to the respiration of organisms and plants in soil, which is affected by temperature. A significant change in the concentration of CO_2 occurs with temperature changes of 0.5 C°or 5% of the actual temperature. Given the sampling period of 10 minutes, we set the learning phase to 1 hour to accumulate at least $m = 6$ samples before applying the prediction technique.

Indoor Sensing (INDOOR). Our next application case study is arguably one of the first publicly available data sets collected from a WSN. As such it has been used by earlier data prediction studies [6], [10], offering direct comparison between our results and prior published results. In this dataset, the light, temperature, humidity, and battery voltage of 54 nodes (Mica2Dot) deployed inside the Intel Berkeley Research Lab are collected. The data trends are dramatically different from those of outdoor WSNs, as the indoor sensors are influenced by artificial factors such as the heating ventilation and air conditioning (HVAC) system and human-controlled lighting.

The dataset covers 36 days, in which the nodes reported 2.3 million values for each of the aforementioned physical quantities. In our experiments, we do not consider voltage as it is highly correlated to temperature. With this data set, we set the tolerance parameters for temperature as in SOIL, and those of the other two quantities as an estimate of the perceivable effect on the comfort of the building occupants.

Water distribution (WATER). In our final case study we consider a simulated sensor network monitoring hydraulic and chemical phenomena in drinking water distribution piping systems. The data comes from EPANET 2.0 [11], an accurate modeling tool that tracks the water flow in each pipe, the water height in each tank, the pressure at each node, and the chlorine concentration throughout the network during a specified simulation period. This dataset has been used in several previous studies (e.g., in [12]–[15]), and contains radically different variation patterns compared to our other datasets. Specifically, while the variations in the latter exhibit a 24-hour diurnal cycle, the variation period in the EPANET data is shorter and harder to model by linear prediction techniques, therefore constituting a worst-case in our context.

From this application, we consider a dataset containing measurements of the chlorine concentration every 5 minutes at 166 junctions in the water distribution network for a 15-day interval, for a total of $715,460$ measurements. This data set exhibits a global, daily periodic pattern following residential demand, and a slight time shift across different junctions, due to the time it takes for fresh water to flow down the pipes from the reservoirs. We assume a value tolerance of (5%,0.0001), which allows sensors measuring very low chlorine concentrations to report data.

4.2 Comparing DBP against the State of the Art

The goal of data prediction is to reduce the transmission ratio without stepping outside the tolerated error values. To evaluate this, we consider all the available data sets described earlier and compare the suppression percentage of DBP to several other techniques from the literature we concisely describe here, and place in a wider context in Section 6:

TABLE 2
Root Mean Squared Error and Suppression Ratio

Application	Dataset	Root Mean Squared Error *				Suppression Ratio(%) *			
		DBP	PLA	SAF	POR	DBP	PLA	SAF	POR
TUNNEL	Light	**18.867**	19.121	20.031	19.307	**99.74**	99.71	99.71	99.09
SOIL	Air Temperature	0.618	**0.613**	0.6196	0.794	**91.83**	91.77	91.79	89.30
	Soil Temperature	0.352	0.352	**0.3495**	0.361	98.80	98.82	**98.83**	97.83
INDOOR	Humidity	**4.494**	4.540	4.528	4.513	**99.50**	99.47	99.48	98.59
	Light	**23.980**	30.981	25.493	31.480	**97.58**	97.10	97.47	96.43
	Temperature	**1.972**	2.130	**1.972**	2.336	**99.60**	99.58	99.59	98.95
WATER	Chlorine	0.008	0.008	0.008	**0.007**	89.81	89.44	89.57	**92.58**

* **Underlined bold-face** numbers denote the lowest RMSE error or the highest suppression ratio.

- *Piecewise Linear Approximation* (PLA) is a popular technique that uses least square error linear segments to approximate a set of values [10]. In our case, each node uses a single segment to model sensed values.
- *Similarity-based Adaptable Framework* (SAF) [6] relies on an autoregressive moving-average model of order 3 with moving-average parameter of order 0. In SAF a value V_i is predicted by a linear combination of the last three: $V_i = l_i + \alpha_1(V_{i-1} - l_{i-1}) + \alpha_2(V_{i-2} - l_{i-2}) + \alpha_3(V_{i-3} - l_{i-3})$, where α_1, α_2, α_3 are constants the model must estimate, and l_i models the linear trend of data over time.
- *Polynomial Regression* (POR). In contrast to DBP, POR allows the use of non-linear models for prediction. Intuitively, this may yield better performance through a better fit to the data. Like PLA, POR uses the least squares measure for selecting the most appropriate coefficients for the polynomials, which have the form $y = \sum_{k=0}^{p} \alpha_i x^i$. For this study, we evaluated polynomials of order $p = 2, 3, 4$, but show only $p = 2$ as it provides the best results for POR.

We used the value and error tolerances matching each target application as outlined in Table 1. The duration of the learning window m is the same across all techniques, and is also specified in Table 1. Finally, for DBP we used $l = 3$ edge points; this value yields the best performance, although its impact is nonetheless rather limited.

First we consider the error of predicted vs. actual sensor values. Like other studies [10], [12], we use Root Mean Squared Error (RMSE) as an indicator of the quality of the predicted time series at the sink. We define it as $RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(V_i - \hat{V}_i)^2}$ where V_i and $\hat{V}_i$ are the sensed and predicted values, respectively, and N is total number of values sensed. Table 2 shows the RMSE across all data sets. DBP minimizes the error in 4 out of 7 datasets and is the second best in the others, confirming its ability to accurately predict the sensed values. This result is impressive, considering that the approaches we are comparing against, unlike DBP, are expressly designed to reduce RMSE.

All approaches perform well in terms of data suppression, but DBP achieves the best overall results in 5 out 7 datasets. Table 2 shows that DBP suppresses 99.7% of the message reports in TUNNEL, followed by 99.6% and 99.5% for the temperature and humidity INDOOR datasets.

On the other hand, the chlorine dataset in WATER is characterized by non-linear periodic trends, that are of course better approximated by the polynomial regression function of POR, rather than by linear approximations as in DBP. Indeed, we chose this dataset as a sort of stress test for our technique. Although POR suppresses 2.77% more data reports than DBP, the performance of the latter is still very good, considering that: *i)* DBP is operating outside of its assumptions *ii)* DBP still outperforms both PLA and SAF *iii)* DBP's implementation is significantly easier and less memory-hungry than the other techniques, and therefore easier to integrate on resource-scarce WSN devices *iv)* POR exhibits the worse performance in *all* other datasets, up to 2.53% less reports suppressed w.r.t. DBP in SOIL.

Table 2 shows the *aggregate* data suppression rate, but of course different nodes enjoy different suppression rates, depending on the specific trends they observe in the sensed phenomena. Figure 4(a) provides a concrete view of this statement in WATER, our worst-case dataset, by showing the suppression ratio of each node. Figure 5 provides a more intuitive view on the same dataset by plotting the cumulative distribution function (CDF): a point on the curve represents the *number* of nodes, on the x-axis, that have a suppression ratio less than or equal to the one on the y-axis. The charts confirm the non-uniformity of data suppression, and show again that POR is consistently more efficient at suppressing reports than the other techniques, a consequence of the particular non-linear nature of the WATER dataset, as already mentioned. The lack of detailed information about the deployment of nodes and the trends of the physical phenomena observed, and the inability to run specific tests, prevents us from providing more in-depth observations in WATER, as well as SOIL and INDOOR.

On the contrary, in TUNNEL we do have all the information above. Fig. 4(b) shows the data suppression

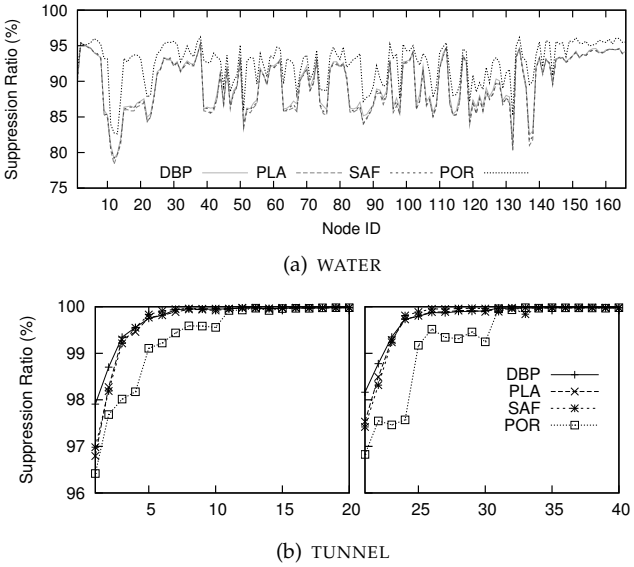


Fig. 4. DBP vs. state-of-the-art techniques on individual nodes in WATER and TUNNEL.

rate for the individual nodes in the tunnel. The chart is split in two to remind the reader that the deployment is constituted of two parallel lines of WSN devices, arranged as shown in Figure 3. Indeed, the node placement motivates the difference in performance among the various nodes. The nodes in the tunnel interior are only marginally affected by the outside lighting conditions; the light data they sense is determined by the rather constant illumination provided by the tunnel lamps. All data prediction techniques are very effective in this case. On the other hand, the nodes near the entrance are subject to variations in light that can be also quite abrupt (e.g., upon sunrise) and that, contrary to the WATER dataset, are consistently predicted less effectively by POR.

4.3 DBP in Action

In this section we take a closer look at the operation of DBP, showing that our technique can satisfy the error and delay tolerance requirements set by applications. We focus most of our discussion on the TUNNEL dataset, as it is the one for which we have most information, and occasionally compare with the WATER dataset, which is the worst case for DBP.

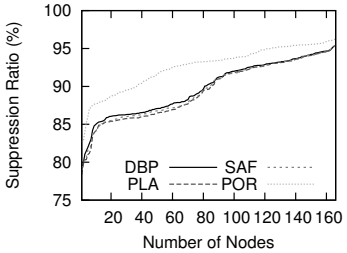


Fig. 5. A different view on Figure 4(a): CDF of suppression ratio for the individual nodes in WATER.

We begin by analyzing DBP in the small, dissecting the operation of a single node over a single day of operation for both these datasets, as shown in Fig. 6. In WATER, we chose node 1 because it has an average suppression ratio w.r.t. other nodes in the same deployment. In TUNNEL, we choose node 1 because, as shown in Fig. 3, it is placed at the tunnel entrance where, in comparison with nodes in the interior, most of the changes in light readings occur. The top charts in Fig. 6 show the values sensed by these nodes both in the original case where data is reported periodically (every 5 minutes for WATER and 30 s for TUNNEL, according to Table 1) and when DBP is applied. In the latter case, the cross points indicate the generation of a new model, while the lines between the points show the values automatically calculated at the sink from those models. The two datasets exhibit different trends: while the values in TUNNEL reflect the light changes induced by sunrise and sunset, the values in WATER are affected only by the concentration of chlorine, which is set arbitrarily by the simulation from which the dataset is extracted. Nevertheless, the charts show that DBP is able to predict very closely the actual values in both cases, while suppressing the majority of messages. For instance, in TUNNEL $2,880$ messages are sent without DBP, against only 25 messages with DBP: a suppression ratio of 99.13%.

As expected, the majority of the DBP models are generated in correspondence of slope changes in the value trends. Interestingly, in TUNNEL these are almost all concentrated around sunrise and sunset: the rest of the time, DBP generates very few models. These observations are confirmed on a global scale by Fig. 7, where we show the overall number of models generated by *all* the nodes in TUNNEL, over time. To measure this, we divide our 24-hour experiment into 5-minute intervals and count the number of models generated by all nodes in each interval. The number of models in any 5-minute interval reaches a peak of 10 after sunrise, a second peak of 4 around sunset, and remains well below this value during the rest of the day. At night, many intervals are present in which no models are generated.

Finally, the bottom charts in Fig. 6 focus again on individual nodes as representative examples, to analyze the error in the values provided by DBP to the application. The solid line indicates the value tolerance set by our application requirements—$\varepsilon_V = (5, 0.0001)$

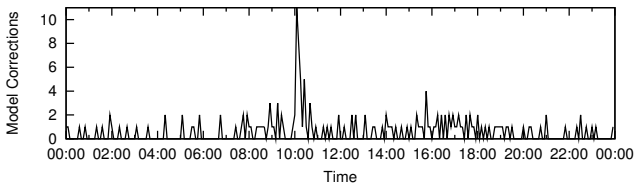
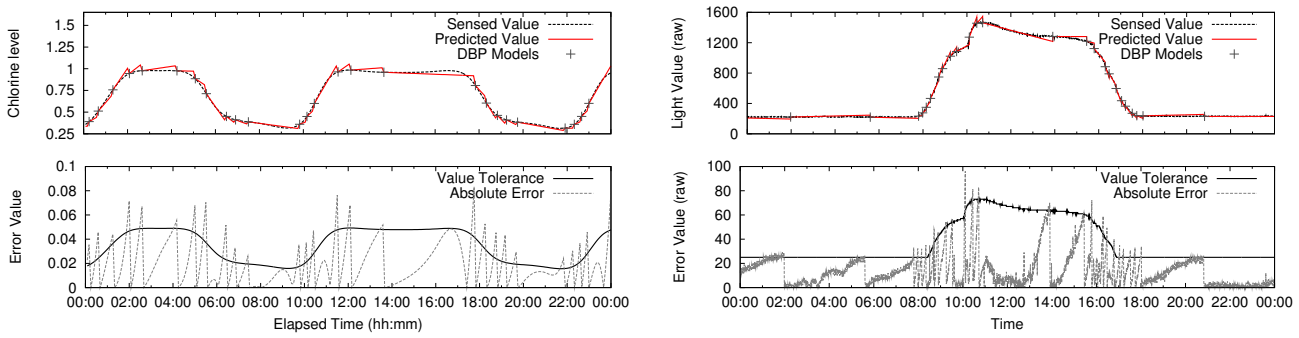


Fig. 7. Total number of DBP model updates in TUNNEL over 5-minute intervals, during a 24-hour experiment.



(a) WATER (b) TUNNEL

Fig. 6. Absolute values (top) and error (bottom), from the WATER and TUNNEL applications with DBP.

in WATER and $\varepsilon_V = (5, 25)$ in TUNNEL—while the lighter line shows the error of DBP as the difference between the predicted value and the sensed value. In most cases, the error falls below the value tolerance. Excursions above the value tolerance are caused by data predicted at the sink that, albeit incorrect, are within the time tolerance. In each of these cases, either subsequent values fell back below value tolerance or a new model was generated after the maximum number of incorrect reports ($\varepsilon_T = 2$ in our case) was exceeded. Interestingly, in many cases (e.g., at night in TUNNEL) one can see the absolute error growing for a while, then dropping and growing again. The drop in error corresponds to the generation of a new model, visible also in the top charts of Fig. 6. The growing error is because the DBP model is linear with a small, but non-zero slope, which is slightly off the measured light values that remain mostly constant. It is also worth noting that, in TUNNEL, the value tolerance at night is dominated by the absolute error ϵ^{abs}, while during the day it is dominated by the relative error ϵ^{rel}. Indeed, the light at the entrance of the tunnel at night amounts to only a few lux, while during the day it can easily exceed a thousand lux. This disparity motivates the use of two different value tolerances.

4.4 Impact of Error Tolerance

The previous evaluation shows that DBP performs well on our datasets, representative of real-world applications. However, we want to explore the parameter space for DBP, to understand the effect of changes in the value and time tolerances on the transmission ratio. To this end, given the high number of combinations, we restrict ourselves to TUNNEL because, as already discussed earlier, for this we have more application information enabling us to better interpret the impact of error tolerances. Figure 8(a)–8(c) show how SR changes at individual nodes of the tunnel for various combinations of parameters. Recall from Fig. 3 that nodes 1–20 are placed on the same North wall, while nodes 21–40 belong to the South wall. We plot a line connecting the SR at each node, because this best highlights the trends as one proceeds from the entrance to the interior of the tunnel (e.g., from node 1 to 20 on the North wall).

In Fig. 8(a) we vary the relative error ϵ^{rel} from 1% to 25%, keeping the absolute error constant $\epsilon^{abs} = 25$. By setting the time tolerance to $\varepsilon_T = 0$, we force all deviations from the value tolerances to be reported. To put these values in context, recall that the value tolerance ε_V is defined as the maximum between the relative and absolute errors, ϵ^{rel} and ϵ^{abs}. In Fig. 8(b) we fix $\epsilon^{rel} = 5\%$ and vary ϵ^{abs} between 0 and 50, keeping $\varepsilon_T = 0$. In Fig. 8(c), we use the value tolerance $\varepsilon_V = (5, 25)$ of our target application and vary ε_T between 0 and 4, i.e., from 0 to 2 minutes.

In all cases it is worth noting that, as expected, the biggest savings are harvested from the nodes inside the tunnel, where light variations are more rare, and absolute values of illuminance are smaller. Under these conditions, the linear nature of DBP accurately models the linear nature of the data.

Interestingly, the trends seen for nodes 21-24 in Fig. 8(a) are due to the flickering of a light that introduced noise to the sensor readings. Nevertheless, even in this case DBP achieved suppression ratios greater than 95% for these nodes. Further, in Fig. 8(b), we clearly see the need for both the absolute and relative value tolerances, as when the error tolerances are very low, e.g., $\epsilon^{abs} = 0$ or $\epsilon^{abs} = 10$, SR is off the bottom of the charts. This is because the light sensors themselves have an error that often takes them outside the small, fixed relative error $\epsilon^{rel} = 5\%$, triggering unpredictable model changes. Further, the flickering light introduces additional noise that DBP cannot compensate for with low error thresholds.

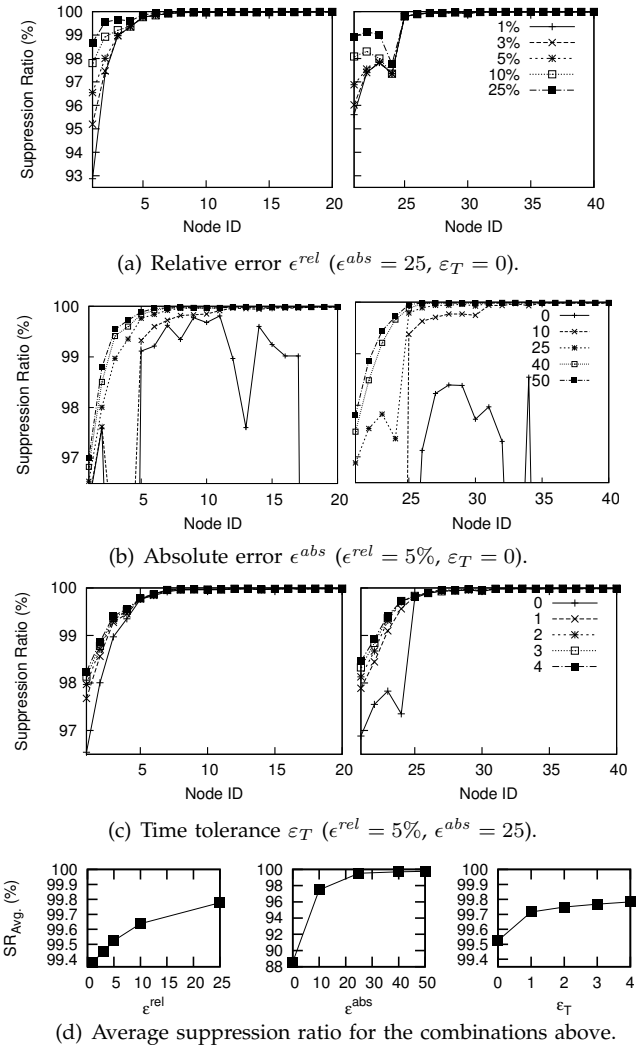
For each of these parameter combinations we also show, in Fig. 8(d), the average SR over all nodes. An increase in the value of ϵ^{rel} brings a near linear increase of SR. Instead, ϵ^{abs} and ε_T both achieve the greatest benefit at small values, with diminishing returns as the value increases. In the former case, the increase in SR progresses rapidly as ϵ^{abs} varies from 0 to 10, going from a suppression ratio of 88% to 98%; a further (and larger) ϵ^{abs} increase from 10 to 25 yields only an additional 2% increase of SR. Similarly, time

tolerance reflects the fact that changes in light values are gradual, and thus introducing even a small delay $\varepsilon_T = 1$ achieves most of the possible gain.

In addition to the combinations in Fig. 8, we also computed the SR achieved with the *strictest* combination of the three parameters: $\epsilon^{rel} = 1\%$, $\epsilon^{abs} = 0$, and $\varepsilon_T = 0$. Even with these worst-case requirements DBP still suppresses, on average, 63% of the reports. More interesting is the *real* combination of parameters ($\epsilon^{rel} = 5\%$, $\epsilon^{abs} = 25$, and $\varepsilon_T = 2$) suggested by the TUNNEL engineers, and used in the rest of our experiments. In this case, the average suppression rate is a staggering 99.7%—SR is increased by almost two orders of magnitude w.r.t. reporting all raw values.

5 A System-wide Evaluation

We now shift our focus from the application layer to the overall system, assessing the impact of data prediction on the full WSN network stack. As observed in the previous section, all the data prediction techniques studied achieve good results with all data sets, resulting in extremely low, aperiodic traffic. Therefore, to study the system as a whole, we can restrict our evaluation to a single, representative application. We chose the TUNNEL one, for which we have full access not only to the dataset, but also to the software deployed, and the related operational environment.



(a) Relative error ϵ^{rel} ($\epsilon^{abs} = 25$, $\varepsilon_T = 0$).

(b) Absolute error ϵ^{abs} ($\epsilon^{rel} = 5\%$, $\varepsilon_T = 0$).

(c) Time tolerance ε_T ($\epsilon^{rel} = 5\%$, $\epsilon^{abs} = 25$).

(d) Average suppression ratio for the combinations above.

Fig. 8. Tunnel: Impact of error tolerance parameters on suppression ratio.

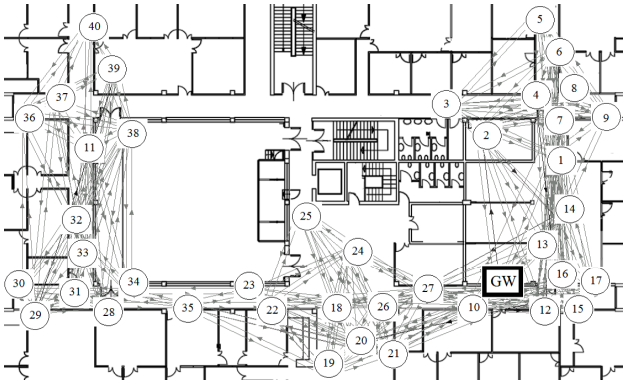
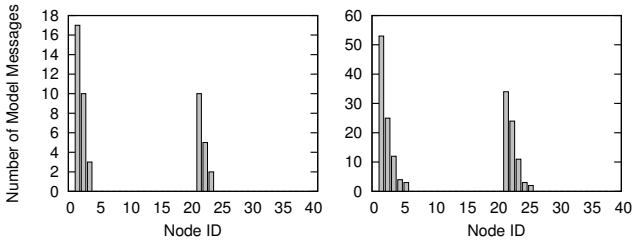


Fig. 9. Testbed map and connectivity.

Our evaluation considers the combination of DBP and a mainstream WSN network stack composed of CTP [3], BoX-MAC [4], and TinyOS v2.1.1. We experiment in two settings: an operational road tunnel to evaluate DBP in the real conditions of our target application, and an indoor testbed, representative of scenarios with different connectivity.

Tunnels are complex environments where factors such as road traffic affect network behavior. For example, we previously observed [16] that in the presence of high traffic, nodes consistently select parents on their same side of the tunnel, while at low traffic nodes across the tunnel are often selected. This is due to the interference caused by vehicles, nevertheless, it profoundly affects the shape and maintenance cost of the routing tree. For these experiments, we relied on the 40-node WSN in Fig. 3. The testbed is composed of 40 TelosB nodes in a 60x40 m^2 office area shown in Fig. 9. The node placement, along with the power setting of -1 dBm, creates a network topology that approximately forms three segments, reminiscent of the linear tunnel topology, but with larger diameter.

To assess directly the impact of the network stack on the improvements theoretically attainable by DBP, we "replayed" the same data we used in Section 4 both in the tunnel and testbed. As we could not re-execute the entire 47-day data set with multiple combinations of parameters, for the tunnel we selected a single 23-hour period, ensuring variability in the vehicular traffic. Moreover, restrictions on the usage of the testbed forced us to run experiments only for a few hours. Therefore, in this latter case we chose to focus on the sunrise period, the most challenging because values change dramatically and, unlike sunset, are not followed by constant light levels at night. Fig. 10 shows the number of models generated by each node in both cases. We begin the evaluation after DBP



(a) Testbed: $SR = 99.52\%$ (2 h). (b) Tunnel: $SR = 99.86\%$ (23 h).

Fig. 10. Number of model update messages.

has been initialized, specifically after generation and transmission of the first model.

We next consider how application data delivery, network lifetime, and routing costs are affected by DBP, with the goal of understanding if improvements can be achieved by coordinating its functionality with the layers below it. All these metrics are deeply affected by the operation of the MAC layer, specifically the rate at which the radio duty cycles, which therefore becomes a key parameter in our experiments. At low sleep intervals, nodes frequently check the channel but find no activity, increasing idle listening costs. On the other hand, with long sleep intervals, the cost to transmit a packet increases. Specifically in BoX-MAC, transmission to a non-sink node takes on average half the sleep interval, due to the fact that the sender must transmit until the receiver wakes up, receives the packet, then acknowledges its reception [4]. Long transmission times also increase the probability of packet collisions among hidden terminals, further decreasing the delivery ratio and increasing energy consumption. The ideal sleep interval balances idle listening and active transmission costs. To identify the best interval for the tunnel application, we ran experiments with a range of values from 500 to 3000 ms.

5.1 Data Delivery

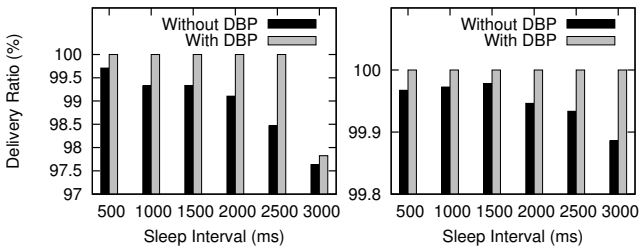
DBP greatly reduces the amount of data in the network w.r.t. the baseline where all nodes send data every 30 s. The reduction in data transmitted reduces the probability of collisions, therefore increasing the delivery ratio. This is evident in Fig. 11, where the system with DBP loses fewer messages than without DBP. In all cases the delivery is very good, above 97%, but DBP actually achieves 100% in all cases and in both scenarios, except for the case with the maximum sleep interval of 3000 ms in the testbed. In this case, a single model message was lost; however, as the absolute number of model changes is small, the total delivery ratio drops by almost 3%. Although this loss rate may be acceptable without DBP, losing a single DBP model has the potential to introduce large errors at the sink, as the latter will continue to predict sensor values with an out-of-date model until the next one is received. This suggests that, based on the target environment, dedicated mechanisms may be required to ensure reliability of model transmissions.

5.2 Lifetime

It is well-known that the radio is a power-hungry component. Therefore, we measure its duty cycle, as the time spent in communication is the most significant factor contributing to system lifetime. Fig. 12 clearly shows that DBP enables significant savings at any sleep interval. Without DBP, the optimal sleep interval yielding the lowest duty cycle is 1500 ms. Further increasing the sleep interval decreases the idle listening cost, but it increases the transmission cost as the average transmission duration is half the sleep interval. This phenomenon instead bears a negligible effect in DBP where transmissions are greatly reduced. In this case, longer sleep intervals can be used to increase lifetime without affecting data delivery.

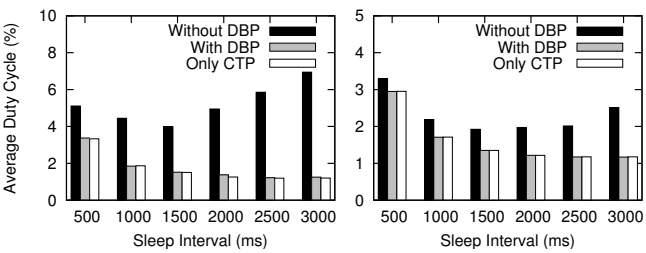
Fig. 12(a) shows that in the testbed, with a sleep interval of 1500 ms (i.e., the best without DBP), DBP yields more than twice the lifetime of the no-DBP baseline—i.e., the WSN running DBP lasts twice as long, with the same MAC settings. Using the best sleep interval in both cases (i.e., 1500 and 3000 ms, respectively) yields a three-fold lifetime improvement. The energy savings in the tunnel, in Fig. 12(b), are less remarkable although still significant. The network diameter in the tunnel is much smaller w.r.t. the testbed, due to the waveguide effect described in [16]; many direct, 1-hop links to the sink exist, leaving less room for improvement.

The impact of 1-hop links to the sink is worth commenting further. Indeed, because the sink is always on, it quickly receives and acknowledges a packet, making transmissions from its direct children very short and therefore low-energy. This can be seen clearly in Fig. 13 where, for the tunnel experiments,



(a) Testbed (2 hours). (b) Tunnel (23 hours).

Fig. 11. Delivery ratio.



(a) Testbed (2 hours). (b) Tunnel (23 hours).

Fig. 12. Duty cycle. The y-axis scale is different.

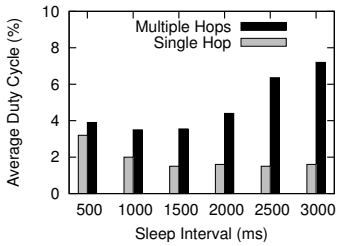


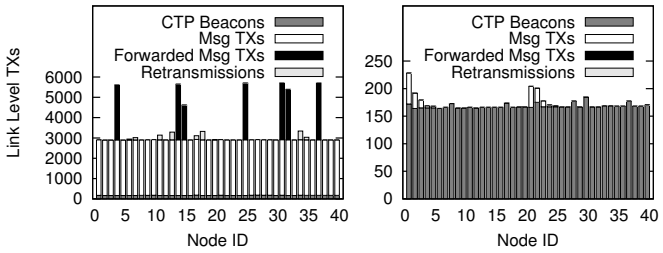
Fig. 13. Tunnel: Duty cycle vs. sink distance, no DBP.

we measure separately the duty cycle of the nodes that spent their entire lifetime directly connected to the sink and those that, at any time, were more than one hop away. Directly-connected nodes enjoy much lower energy costs. The plot considers only the case without DBP. Interestingly, with DBP *all* the nodes reporting model changes (Fig. 10(b)) were in direct range of the sink. Indeed, as shown in Fig. 3, the latter is attached to the gateway placed at the entrance, where light variations, and hence model changes, occur. This placement was not our deliberate choice, as it was originally determined by the available power panels in the tunnel. Nevertheless, it hints at the fact that, if *a priori* knowledge is available about the sensors that are likely to generate the most model variations, this can be exploited when determining the gateway placement. A similar optimization is not possible without DBP, as all nodes must send data.

5.3 Routing Costs

A natural question arises at this point: if DBP suppresses over 99% of the messages, why does the network lifetime increase "only" three-fold? This is due to the costs of the network stack, in particular the idle listening and average transmission times of the MAC protocol, and to the overhead of the routing protocol to build and maintain the data collection tree. As we already evaluated the impact of the MAC layer, here we turn to the routing layer.

To isolate the inherent costs (e.g., tree maintenance) of CTP, we ran experiments with no application traffic. The corresponding duty cycle is shown as *Only CTP* in Fig. 12; interestingly, the DBP cost is very close to the cost of CTP tree maintenance, regardless of the sleep interval. A finer-grained view is provided by Fig. 14,



(a) Without DBP. (b) With DBP.

Fig. 14. Tunnel: Total link-level transmissions for a sleep interval of 1500 ms. The y-axis scale is different.

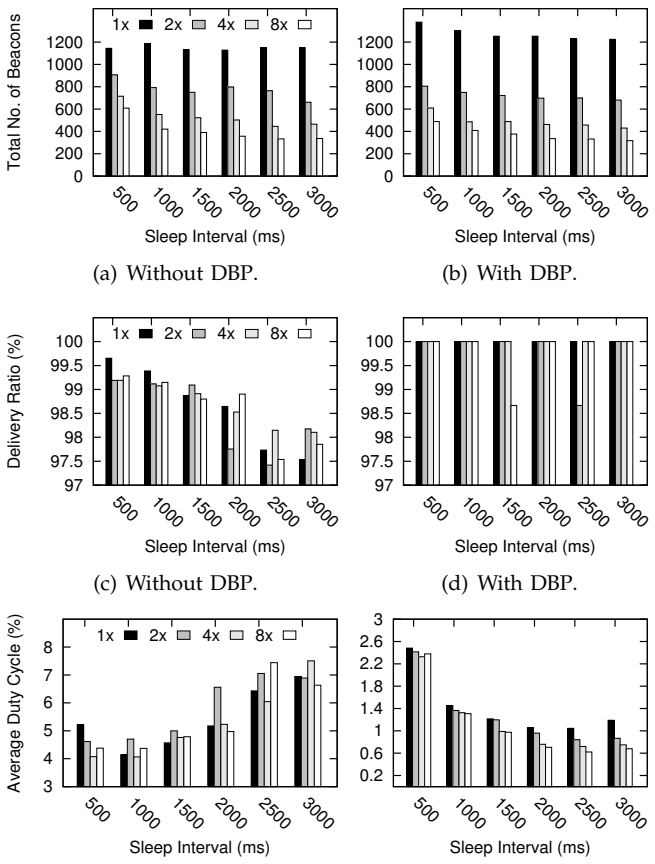
where we analyze the different components of traffic in the network. Without DBP, the dominating component is message transmission and forwarding; significant retransmissions are present for some nodes, while the component ascribed to CTP (i.e., the beacons probing for link quality) is negligible. When DBP is active, the number of CTP beacons remains basically unchanged. However, because application-level traffic is dramatically reduced, CTP beacons become the dominant component of network traffic.

5.4 Cross-layer Routing Optimizations

The routing cost analysis above reveals that beacons, not application traffic, dominate the network traffic and therefore are a limiting factor of the system lifetime. In CTP, the number of beacons, and therefore the cost of beaconing, is determined by an application of the Trickle algorithm [3], which sends one beacon at a random moment within a given time interval. The time interval is initially small (0.125 s by default) to allow CTP to obtain and rapidly propagate accurate link cost estimates. However, to limit beaconing cost, if no link variations are detected the interval doubles, eventually reaching a large maximum value, configured to 500 s by default. When beacons are triggered due to link cost variations, the interval shrinks back to the minimum, then gradually increases back to the maximum. In mainstream application environments, CTP spends most of the time at the maximum beacon interval. Here, we investigate how to reduce the beaconing cost, yet allowing CTP to function properly in the presence of link variations. To this end, we maintain the core Trickle mechanism but, in addition to the default 500 s, we experiment with larger maximum beacon intervals of 1000, 2000, and 4000 s. We hereafter refer to these values as 1x, 2x, 4x, and 8x.

The experiments we report here are performed in our testbed. However, they are longer than those reported earlier in this section, because CTP requires more time (about 2 hours at 8x) to reach a larger maximum beacon interval. The 4-hour duration of experiments is determined by restrictions on the testbed usage, and the need to keep the total experiment time manageable under the many combinations of parameters under consideration. Further, this set of experiments was also run at a later time w.r.t. those we presented earlier, and originally in [2]. Since then, the environment where the testbed is deployed underwent changes (e.g., a few walls were moved) that, albeit minor, affected connectivity.

The experiments we present here, therefore, are also the opportunity to validate our earlier results on a slightly different WSN setup and longer experiment duration. Fig. 15 shows our results, with different combinations of maximum beacon intervals and MAC-level sleep intervals. A comparison with Fig. 11–12 can be easily seen by focusing on 1x,



(a) Without DBP. (b) With DBP.

(c) Without DBP. (d) With DBP.

(e) Without DBP. (f) With DBP.

Fig. 15. Testbed (4 hours): beacon transmissions, delivery ratio, and duty cycle for different combinations of sleep intervals and beacon intervals. Note the different scale in the bottom charts for duty cycle.

the default maximum beacon interval. The trends are very similar to those observed earlier, with two major differences. First, the optimal sleep intervals without and with DBP are now 1000 ms and 2500 ms, respectively. Second, a comparison of the duty cycle for these optimal values shows a 4-fold improvement. The new setting is therefore more advantageous for DBP than in Section 5.2, where we obtained "only" a 3-fold improvement. Interestingly, our new setting is therefore a more challenging scenario for our goal of exploiting cross-layer routing optimizations to improve further over what DBP can achieve alone.

Looking at the rest of Fig. 15 we note that, as expected, the total number of beacons decreases as we increase the maximum beacon interval, yielding up to 70% fewer beacons at 8x. The number of beacons remains essentially the same with and without DBP, as shown in Fig. 15(a)–15(b). This is an expected consequence of this metric being tied to the stability of the network rather than the data traffic. Moreover, the impact of data delivery is dominated more by the MAC sleep interval than the maximum beacon interval, as shown in Fig. 15(c)–15(d). The trends are similar to those in Fig. 11(a), where the system with DBP remains at 100% except for two cases where a single packet loss occurs, while the delivery without DBP degrades as the sleep interval increases.

On the other hand, increasing the maximum beacon interval bears a dramatic effect on lifetime. Without DBP, beacons are a small percentage of the overall network traffic. Therefore, as Fig. 15(e) shows, the effect of increasing the beacon interval does not bear a definite effect on duty cycle. Instead, with DBP this *always* provides a benefit. In particular, the duty cycle at the optimal sleep interval of 2500 ms is reduced by 40% when moving from 1x to 8x. If we compare this optimal DBP configuration (2500 ms, 8x) to the optimal configuration without DBP (1000 ms, 1x), the lifetime of the former is *seven times higher* than the latter. These results confirm that cross-layer tuning of the MAC-layer sleep interval and the routing layer beacon interval can lead to significant improvements.

Finally, we note that even with 4-hour experiments, the time for CTP to ramp up to the longest beacon interval is a significant fraction of the total experiment time, from 17 minutes at 1x to 135 minutes at 8x. Therefore, we expect the positive results shown here to be a conservative measure of the gains that can be attained in real deployments, where the effect of infrequent beacons are predominant in the long term.

6 Related Work

The limited resources, variable connectivity, and spatio-temporal correlation among sensed values make efficiently collecting, processing and analyzing WSN data challenging. Early approaches use in-network aggregation to reduce the transmitted data, with later approaches addressing missing values, outliers, and intermittent connections [17]–[19].

Data prediction has also been extensively studied. Probabilistic models [20], [21] approximate data with a user-specified confidence, but special data characteristics, such as periodic drifts, must be explicitly encoded by domain experts. In a similar parametric approximation technique [22], nodes collaborate to fit a global function to local measurements, but this requires an assumption about the number of estimators required to fit the data. In contrast, DBP requires neither expert domain knowledge nor lengthy training, but provides hard accuracy guarantees on the collected data. PAQ [23], SAF [6], and DKF [24] employ linear regression, autoregressive models, and Kalman filters respectively for modeling sensor measurements, with SAF outperforming the others.

As an alternative to data modeling, some solutions seek to suppress reporting at the source by using spatio-temporal knowledge of data [25] or by identifying a set of representative nodes and restricting data collection to it [26]–[30]. Others take the remaining energy of individual nodes [31] into account. These approaches further reduce communication costs

and can be applied in combination with DBP. Work on continuous queries for data streams studies the tradeoff between precision and performance when querying replicated, cached data [32]. Finally, several studies focus on summarizing streaming time series, showing that the choice of the summarization method does not greatly affect the accuracy of the summary. In our experiments, we compared against PLA [10], as it can be efficiently computed.

The above data driven approaches have been evaluated theoretically, but no prior work explores the real effect of the network stack on the overall energy savings. Network-level energy savings approaches can be classified into MAC level, cross-layer, or traffic-aware.

At the MAC layer [33], low-power listening protocols such as BoX-MAC [4] dominate real deployments due to their availability, simplicity and effectiveness in reducing duty cycle. Nevertheless, as our analysis shows, parameters such as the listening interval must be carefully tuned.

Vertical solutions crossing network layers achieve extremely low duty cycles. Dozer [34] achieves permille (0.1%) duty cycle by taking a TDMA-like approach in which a tree parent autonomously schedules its transmissions to and from its children. Unfortunately, Dozer does not scale well and is prone to choose poor quality parents. Koala [35] achieves similar low duty cycles, but by explicitly accepting delays between data generation and delivery. Koala is characterized by long periods of very low-power local data sampling followed by brief, high-consumption data collection intervals. While the energy savings are significant, the significant delays are not acceptable for closed-loop systems like our tunnel.

Other techniques [36], [37] adapt sleep schedules according to traffic statistics. Unfortunately, the data modeling approaches outlined above, of which DBP is another example, are difficult to predict due to the variability of the application data itself and the interaction with the modeling technique.

7 Conclusions

Data prediction relies on the fact that many applications can operate with approximated data, as long as the difference w.r.t. the real one remains within certain limits. In these cases, WSN nodes can avoid reporting *all* sensed data, communicating only the deviations.

We proposed a new technique called DBP and, based on more than 13 million data points from 4 real-world WSN applications showed that it suppresses up to 99% of the message reports, often performing better than other approaches. DBP is also considerably simpler than the competition, posing minimal demands on resource-scarce WSN devices. The *practical* usefulness of DBP is reinforced by our system-wide evaluation on real-world WSN deployments, a lab testbed and a road tunnel, showing that DBP can significantly improve lifetime: by tuning jointly the MAC and routing layers, we obtain a 7-fold improvement w.r.t. mainstream periodic reporting.

Our results suggest a few conclusions. First, further reduction in data traffic would have little practical impact on the WSN lifetime, as network costs are dominated by control operations rather than data forwarding. Therefore, to further improve lifetime, we must address the extremely low data rates resulting from data prediction techniques, by considering radically different network stacks. Second, while a certain amount of loss is usually tolerable, when data prediction is used the loss of a single data model may significantly increase the error of data used by the application. Therefore, reliable mechanisms, beyond those of most routing protocols, should be considered.

Acknowledgments

The authors thank the Autonomous Province of Trentino that partially funded this work within the TRITon project. These activities have also been partially funded through the EU Cooperating Objects Network of Excellence (CONET—FP7-2007-2-224053), and by the European Institute of Innovation & Technology (EIT ICT Labs—Activity 12175 and 12149).

References

[1] T. Palpanas, "Real-time data analytics in sensor networks," in *Managing and Mining Sensor Data*, C. Aggarwal, Ed. Springer, 2012.

[2] U. Raza, A. Camerra, A. Murphy, T. Palpanas, and G. Picco, "What Does Model-driven Data Acquisition Really Achieve in Wireless Sensor Networks?" in *Proc. of the 10^{th} IEEE Int. Conf. on Pervasive Computing and Communications (PerCom)*, 2012.

[3] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis, "The collection tree protocol," in *Proc. of the Int. Conf. on Embedded Networked Sensor Systems (SenSys)*, 2009.

[4] D. Moss and P. Levis, "BoX-MACs: Exploiting Physical and Link Layer Boundaries in Low-Power Networking," Stanford Information Networks Group, Tech. Rep. SING-08-00, 2008.

[5] M. Ceriotti, M. Corrà, L. D'Orazio, R. Doriguzzi, D. Facchin, S. Guna, G. P. Jesi, R. L. Cigno, L. Mottola, A. L. Murphy, M. Pescalli, G. P. Picco, D. Pregnolato, and C. Torghele, "Is there light at the ends of the tunnel? Wireless sensor networks for adaptive lighting in road tunnels," in *Proc. of the Int. Conf. on Information Processing in Sensor Networks (IPSN)*, 2011.

[6] D. Tulone and S. Madden, "An energy-efficient querying framework in sensor networks for detecting node similarities," in *Proc. of the Int. Conf. on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM)*, 2006.

[7] J. Polastre, R. Szewczyk, and D. Culler, "Telos: Enabling ultra-low power wireless research," in *Proc. of the Int. Conf. on Information Processing in Sensor Networks (IPSN)*, 2005.

[8] Life Under Your Feet Project, lifeunderyourfeet.org/en/src/.

[9] J. Gupchup, A. Terzis, R. Burns, and A. Szalay, "Model-based event detection in wireless sensor networks," *arXiv preprint arXiv:0901.3923*, 2009.

[10] T. Palpanas, M. Vlachos, E. J. Keogh, and D. Gunopulos, "Streaming time series summarization using user-defined amnesic functions," *IEEE Trans. on Knowledge Data Engineering*, vol. 20, no. 7, 2008.

[11] EPANET, www.epa.gov/nrmrl/wswrd/dw/epanet.html.

[12] S. Papadimitriou, J. Sun, and C. Faloutsos, "Streaming pattern discovery in multiple time-series." in *VLDB*, 2005, pp. 697–708.

[13] J. Berry, W. E. Hart, and C. A. Phillips, "Sensor placement in municipal water networks," *J. Water*, vol. 131, 2003.

[14] S. Papadimitriou, F. Li, G. Kollios, and P. S. Yu, "Time series compressibility and privacy," in *VLDB*, 2007, pp. 459–470.

[15] J. Sun, S. Papadimitriou, and C. Faloutsos, "Online latent variable detection in sensor networks," in *ICDE*, 2005.

[16] L. Mottola, G. Picco, M. Ceriotti, S. Guna, and A. Murphy, "Not All Wireless Sensor Networks Are Created Equal: A Comparative Study On Tunnels." *ACM Trans. on Sensor Networks (TOSN)*, vol. 7, no. 2, 2010.

[17] L. Gruenwald, M. S. Sadik, R. Shukla, and H. Yang, "DEMS: a data mining based technique to handle missing data in mobile sensor network applications," in *Proc. of the Int. Conf. on Data Mgmt. for Sensor Networks (DMSN)*, 2010.

[18] A. Deligiannakis, Y. Kotidis, V. Vassalos, V. Stoumpos, and A. Delis, "Another outlier bites the dust: Computing meaningful aggregates in sensor networks," in *ICDE*, 2009.

[19] W. Wu, H.-B. Lim, and K.-L. Tan, "Query-driven data collection and data forwarding in intermittently connected mobile sensor networks," in *Proc. of Int. Conf. on Data Mgmt. for Sensor Networks (DMSN)*, 2010.

[20] A. Deshpande, C. Guestrin, S. Madden, J. M. Hellerstein, and W. Hong, "Model-driven data acquisition in sensor networks," in *Proc. of the Int. Conf. on Very Large Data Bases (VLDB)*, 2004.

[21] D. Chu, A. Deshpande, J. M. Hellerstein, and W. Hong, "Approximate data collection in sensor networks using probabilistic models," in *ICDE*, 2006.

[22] C. Guestrin, P. Bodik, R. Thibaux, M. Paskin, and S. Madden, "Distributed regression: an efficient framework for modeling sensor network data," in *Proc. of the Int. Symp. on Information Processing in Sensor Networks (IPSN)*, 2004.

[23] D. Tulone and S. Madden, "PAQ: Time series forecasting for approximate query answering in sensor networks," in *Proceedings of the European Wkshp. on Wireless Sensor Networks (EWSN)*, 2006.

[24] A. Jain, E. Y. Chang, and Y.-F. Wang, "Adaptive stream resource management using Kalman filters," in *Proc. of the Int. Conf. on Management of Data (SIGMOD)*, 2004.

[25] A. Silberstein, G. Filpus, K. Munagala, and J. Yang, "Data-driven processing in sensor networks," in *Proc. of the Conf. on Innovative Data Systems Research (CIDR)*, 2007.

[26] Y. Kotidis, "Snapshot queries: Towards data-centric sensor networks," in *ICDE*, 2005.

[27] Z. Zhou, S. Das, and H. Gupta, "Connected k-coverage problem in sensor networks," in *Proc. of the Int. Conf. on Computer Communications and Networks (IC3N)*, 2004.

[28] M. C. Vuran, O. B. Akan, and I. F. Akyildiz, "Spatio-temporal correlation: theory and applications for wireless sensor networks," *Computer Networks*, vol. 45, no. 3, 2004.

[29] H. Jiang, S. Jin, and C. Wang, "Prediction or not? An energy-efficient framework for clustering-based data collection in wireless sensor networks," *IEEE Trans. on Parallel Distributed Systems*, vol. 22, June 2011.

[30] M. Hassani, E. Mller, P. Spaus, A. Faqolli, T. Palpanas, and T. Seidl, "Self-organizing energy aware clustering of nodes in sensor networks using relevant attributes," in *Proc. of the Int. Wkshp. on Knowledge Discovery from Sensor Data*, 2010.

[31] R. A. F. Mini, M. D. V. Machado, A. A. F. Loureiro, and B. Nath, "Prediction-based energy map for wireless sensor networks," *Ad Hoc Networks*, vol. 3, 2005.

[32] C. Olston, J. Jiang, and J. Widom, "Adaptive filters for continuous queries over distributed data streams," in *Proc. of the Int. Conf. on Management of Data (SIGMOD)*, 2003.

[33] J. Rousselot, A. El-Hoiydi, and J.-D. Decotignie, "Low power medium access control protocols for wireless sensor networks," in *Proc. of the European Wireless Conf. (EW)*, June 2008.

[34] N. Burri, P. von Rickenbach, and R. Wattenhofer, "Dozer: ultra-low power data gathering in sensor networks," in *Proc. of Int. Conf. on Information Processing in Sensor Networks (IPSN)*, 2007.

[35] R. Musaloiu-E., C.-J. M. Liang, and A. Terzis, "Koala: Ultra-low power data retrieval in wireless sensor networks," in *Proc. of the Int. Conf. on Information Processing in Sensor Networks (IPSN)*, 2008.

[36] X. Ning and C. G. Cassandras, "Dynamic sleep time control in wireless sensor networks," *ACM Trans. on Sensor Networks*, vol. 6, June 2010.

[37] C. J. Merlin and W. B. Heinzelman, "Duty cycle control for low power listening mac protocols," in *Proc. of the Int. Conf. on Mobile Ad-hoc and Sensor Systems (MASS)*, 2008.

Usman Raza is a PhD student at University of Trento and Bruno Kessler Foundation, working on energy efficiency for WSNs and Cyber Physical Systems. He completed his MSc in 2008 with a National Management Foundation Gold Medal from Lahore University of Management Sciences, Pakistan. He received his BS from National University of Computer and Emerging Sciences, Pakistan in 2004.

Alessandro Camerra is a technical specialist at IBM System and Technology Group and a student at the Politecnico of Milan, Italy. Before that he worked as research assistant at the University of Trento, Italy and as visiting researcher at the University of California, Riverside. His publications cover the area of time series indexing and wireless sensor network technologies.

Amy L. Murphy is a research scientist at the Bruno Kessler Foundation-IRST, a research center in Trento, Italy. Prior to this, she worked as an assistant professor at the University of Lugano, Switzerland and the University of Rochester, NY, USA. Her research interests include the design, specification and implementation of middleware systems for wireless environments including wireless sensor networks.

Themis Palpanas is a professor of computer science at the Paris Descartes University, France. Before that he worked at the University of Trento and the IBM T.J. Watson Research Center. He is the author of 8 US patents, three of which are part of commercial products. He has received an IBM Shared University Research award, 3 Best Paper awards, and has been General Chair for VLDB 2013.

Gian Pietro Picco is a Professor and Head of the Department of Information Engineering and Computer Science (DISI) at the University of Trento, Italy. His work spans the research fields of software engineering, middleware, and networking, and is oriented in particular towards wireless sensor networks, mobile computing, and large-scale distributed systems. His scientific awards include the "Most Influential Paper from ICSE'97" he received in 2007 at the Int. Conf. on Software Engineering. He is a member of the editorial board of ACM Trans. on Sensor Networks (TOSN) and IEEE Trans. on Software Engineering (TSE).